



Security Box

Written By: Tatiana

TOOLS:

- [Hot Glue gun & hot glue \(1\)](#)
- [Scissors \(1\)](#)
- [USB A/B cable \(1\)](#)

[For computer-to-Arduino connection](#)

PARTS:

- [Arduino \(1\)](#)
- [Small Breadboard \(1\)](#)
- [Cardboard box \(1\)](#)
- [Resistor 10K \$\Omega\$ \(4\)](#)
- [Resistor 470 \$\Omega\$ \(2\)](#)
- [Pushbutton \(4\)](#)
- [Green LED \(1\)](#)
- [Red LED \(1\)](#)
- [Parallax/Futaba servomotor \(1\)](#)
- [Jumper Wire \(1\)](#)

SUMMARY

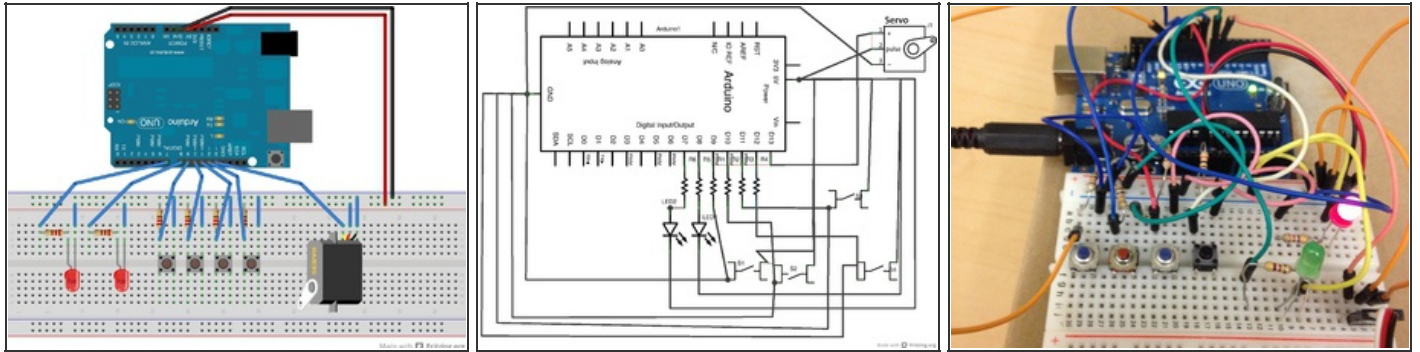
This is a very simple box design using a small box (a shoe box or smaller works) with a simple lock mechanism to represent the opening of the box. When you get the code right, a green LED lights up and the servo moves, unlocking the box (or at least making it easier to open). Otherwise, a red LED is lit up and the box remains locked. The default code is, sequentially, starting with the button attached from pin 12 and going down to the button attached to pin 9.

Step 1 — Set up Box & Servo



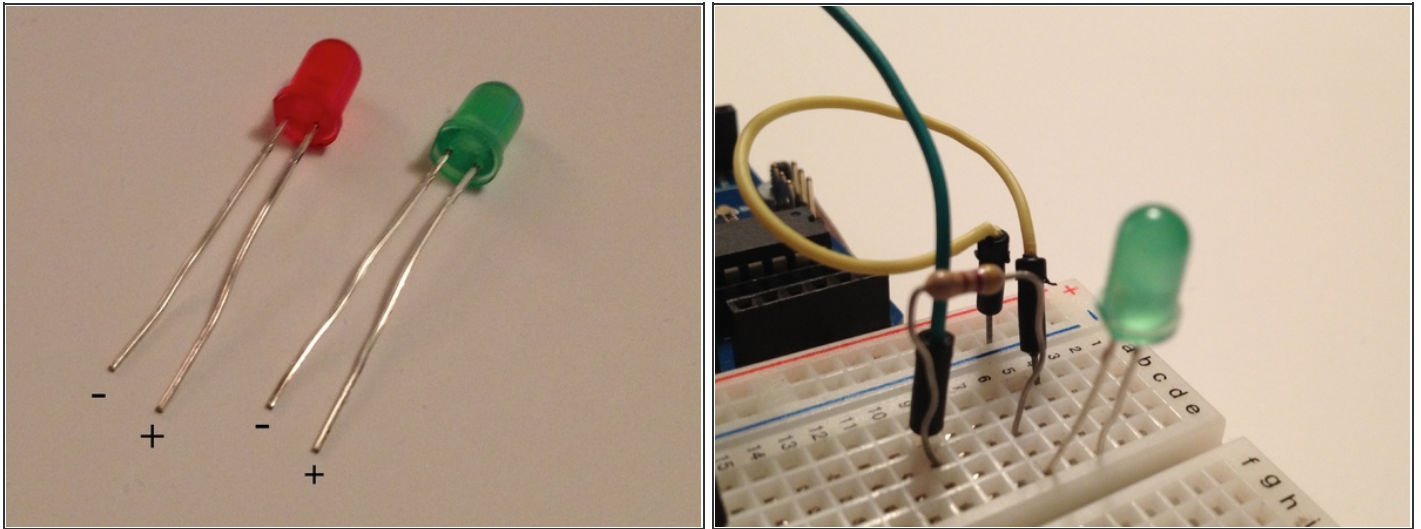
- If your box does not already have a flap-like lid, attach one of the long sides of the lid to that of the box so that it does. (The lid of the box should be attached to the actual box on one side).
- Cut a small piece of cardboard in a "U" shape and then glue it to the top edge of the lid (second picture).
- Cut a piece of cardboard 1-2 inches long and 1/2 inch wide. Fold it in half. Glue this to one spoke of your servo. (You may want to cut the short side farthest from the servo diagonally; it seems to be easier to make the locking mechanism work that way.)
- Tape or glue your servo onto the edge of your cardboard box so that the piece of cardboard is lined up with the "U" shaped loop and can be put inside the loop when closed (see first and third pictures).


Step 2 — Set Up Buttons



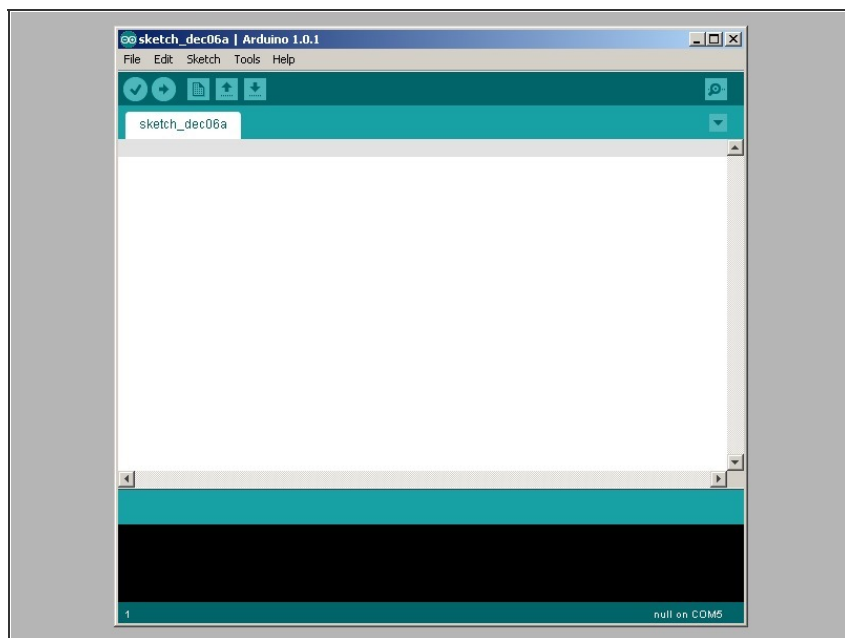
- *Note:* Columns are vertical in the first picture (perpendicular to the indent) and rows are horizontal in the first picture (parallel to the indent). Blue/Black/Negative is for ground. Red/Positive is for the power, 5V.
- Place the pushbuttons so that each one is straddling the indent in the middle of your breadboard. (Buttons will only go into the holes if they're in the proper orientation.)
- Put one lead of a 10K Ω resistor in the same column as one left prong of the button, and put the other lead of the resistor on the red row along the edge. Repeat for each button.
- Between the resistor and the left button prong, in the same column, put a jumper wire going to the pin on the Arduino you want to connect that button to. Repeat for each button. (Use pins 12, 11, 10, and 9 respectively).
- Put one end of a jumper wire in the same column as the right button prong, and the other end in the blue row along the edge. Repeat for each button.
- (I apologize for the messy circuit diagram, I had some trouble with fritzing).

Step 3 — Attach LEDs & Servo



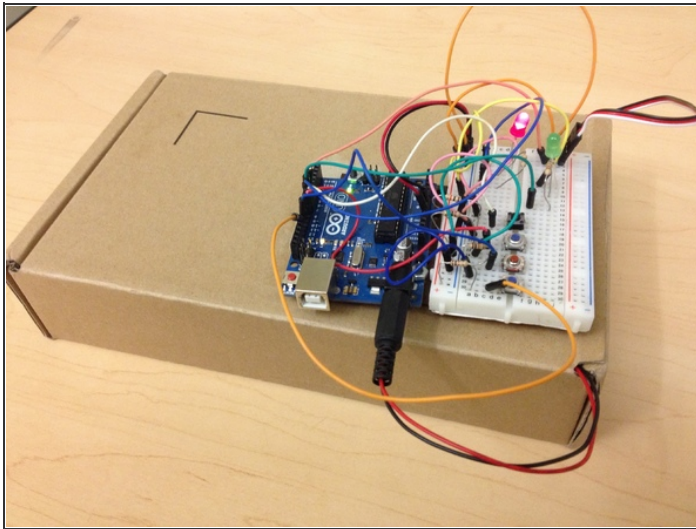
- *Note:* Be careful that the columns you use below are not overlapping with any buttons, etc. 
- Look at the first picture and determine which side of your LEDs (red and green) are positive and negative. (If you look closely the bottom of the LED has a flat side, which corresponds to the negative side of the LED.) Make note of this, and place the LEDs in the breadboard.
- Place a 470 Ω resistor from the positive side of the LED to an unused column. Place a jumper wire from this column to the pin on the Arduino which that LED is assigned to (pin 7 for green, 8 for red).
- Place a jumper wire going from the negative side of each LED to the the blue row.
- Plug your servo into three unused columns on the breadboard. In the same column as the white wire put a jumper wire going to pin 6 on the Arduino. From the red wire put a jumper wire going to 5V. From the black wire put a jumper wire going to ground.

Step 4 — Upload Code



- Open the Arduino program. If you don't have it download it [here](#).
- Copy and paste the code found [here](#)
- Attach your Arduino to your computer (using a USB cable) and upload the code to your Arduino.
- Basically what the code does is store the pin number of the button you pressed in an array (a variable that can store multiple numbers) and it checks each variable in the array each time the code loops through.

Step 5 — Put It Together!



```
    }
    else {
      //The button state is low, reset the lastDebounceTime
      lastDebounceTime[counter] = time;
    }
  }

  if(numberButtonsPressed >= 4) {
    numberButtonsPressed=0;
  }

  //Check if the code is what we want
  if((code[0] == 12) && (code[1] == 11) && (code[2] == 10) && (code[3] == 9)) {
    digitalWrite(greenLedPin, HIGH); //Turn green LED on
    digitalWrite(redLedPin, LOW); //Turn red LED off
    myservo.write(120); //Move the servo to 120 degrees (unlocked)
  }

  //If the code is not what we want
  else {
    digitalWrite(redLedPin, HIGH); //Turn red LED on
    digitalWrite(greenLedPin, LOW); //Turn green LED off
    myservo.write(140); //move the servo to 140 degrees (locked)
  }
}
```

- Place the Arduino on top of the box lid and tape it in place. The wires connected to the servo should go through a crack in the box. (If you can't find a crack to put the wire through, poke a hole in the side of the box.)
- If you want to use your locking box without having it connected to your computer, you can attach a 9V battery to the Arduino and put the battery in the box.
- To change the required button order you just have to change the parameters of the `if` statement on line 122 (the last `if` statement, next to the pink box in picture two).

If you mess up when entering the code to unlock the box, press the reset button on the Arduino since the reset code is finicky.

This document was last generated on 2012-12-12 08:38:50 AM.